whether or not the line contains dirty data. The D-RAMset-Clean command may be performed by examining the valid and dirty bits associated with each line. The D-RAMset-Clean command then copies back to external memory **106** only those lines that have valid and dirty data. In embodiments without dirty bits, the D_RAMset-Clean preferably copies all valid entries from D_RAMset **126** to external memory **106**. The D-RAMset-Flush command invalidates lines within the D-RAMset **126** by clearing the relevant valid bits **237**. The D-RAMset-Clean and D-RAMset-Flush commands may be performed in one of at least three variations. In one variation, the D-RAMset-Clean and D-RAMset-Flush commands perform their respective actions on all of the lines of the D-RAMset **126** (D-RAMset-CleanAll and D-RAMset-FlushAll). In another variation, the D-RAMset-Clean and D-RAMset-Flush commands perform their respective actions on just those lines in the D-RAMset **126** that fall within a range of addresses specified as operands in the commands (D-RAMset-CleanRange and D-RAMset-FlushRange). A third variation permits the D-RAMset-Clean and D-RAMset-Flush commands to act on a single address within the D-RAMset **126** (D-RAMset-CleanEntry and D-RAMset-FlushEntry) providing the corresponding data address to be saved or invalidated.

[**0061**] One or more commands are used to specify whether a data array **238** configured as a RAMset is to function as a Local RAM or as cache. The D-RAMset-policy-set command is used in this regard. In the embodiments described below, this command is implemented as two separate commands called the SPP command and the CP command. Such commands may set one or bits in a register to indicate how a data array **238** is to be used. The bit that is set may comprise a bit in the status register R**15**, the LR/C bit **231** in a register in the cache controller **222** (**FIG. 7**), or in another control register (not specifically shown) in the JSM **102**. Once the bit is set to specify the desired behavior of the associated data array, the cache controller **222** reads the state of the bit to determine the desired allocation policy to implement. The bit may be set, for example, to specify a Local RAM behavior referred to as the scratchpad policy ("SPP") effectuated by the SPP command. In that mode, fetches from external memory are eliminated on cache misses as explained above. Alternatively, the bit may be set so as to specify a cache-based allocation policy in which fetches from external memory are performed on misses before accessing the target data. This latter policy is referred to as the cache policy ("CP") and is effectuated by the CP command. The execution of the SPP and CP commands may be performed dynamically during run-time.

[**0062**] **FIG. 10** illustrates an overflow condition. At **610**, the D-RAMset **126** may comprise local variables **612** associated with a method A and associated metadata **614**. The metadata **614** may comprise various runtime dependent data and local variable pointers as described previously. At **615**, method A invokes method B. In this example, the size of method B's local variables **620** and metadata **622** in memory block **616** is greater than the amount of memory **621** available for allocation to local variables in the current D-RAMset which is identified as $\mathbf{126}_{\text{current}}$. In accordance with some embodiments of the invention, method B's local variables and metadata may be mapped to the two-way set associative cache as explained previously. In accordance with other embodiments, a new memory page may be allocated and mapped onto the D-RAMset **126** such as that depicted in **FIG. 9**. Remapping the D-RAMset **126** may include saving off one or more local variables and metadata from the RAMset's current data. Writing such RAMset data to memory is performed by an operation called a "clean" operation. At any rate, at **625** when method B completes and returns, the JVM **108** preferably re-maps the D-RAMset with method A's local variables and metadata.

[**0063**] Referring now to **FIG. 11**, an overflow condition may be handled as follows and may also be applicable upon a context switch. At **650**, before switching to a new memory page, all local variables and associated metadata from the unfinished method(s) present in the D-RAMset **126** preferably are copied to external memory **106** preferably using the D-RAMset-CleanRange command. As explained above, this command comprises the D-RAMset-Clean command and operands that specify a range of addresses to clean (copy back to external memory **106**). The range of addresses to be cleaned include those addresses from the base address stored in the Full_set_tag register **232** to an address corresponding to the sum of the local variable pointer (PTR LV) **651** and a value **653** that corresponds to the size of the current local variable space. Alternatively, the D-RAMset-CleanAll command could be used to clean the entire D-RAMset. Further still, one or more repetitions of the D-RAMset-CleanEntry command may be performed to clean the desired range. At **652**, a new page preferably is allocated to the D-RAMset **126**. The previous value of the Full_set_tag register **232** is saved in the new metadata stored in the D-RAMset and the Full_set_tag register **232** is reprogrammed with a new page base address.

[**0064**] **FIG. 12** illustrates an underflow condition and may also be applicable upon a context switch. A return from a method may prompt a D-RAMset change to a different page (e.g., a page previously mapped to the RAMset **126**). In general, management of the reloading of the D-RAMset preferably comprises fetching values from external memory **106** only on the first access of each relevant line. As described herein, the JSM **102** includes the ability to dynamically set the load policy. In accordance with preferred embodiments of the invention, this reloading of the D-RAMset underflow situation may be handled as follows. At **660**, the previous value of the D-RAMset base (described above as being stored in metadata from the Full_set_tag register **232**) is retrieved from the D-RAMset's metadata and reloaded into the Full_set_tag register **232**. At **662**, before restoring the previously saved local variable and metadata values, the data in the D-RAMset **126** preferably is invalidated by the D-RAMset-FlushAll command (invalidates the entire D-RAMset). Finally, the D-RAMset allocation policy is configured to the cache policy by the CP command to permit fetches to occur from external memory **106** the first time an access to a particular line is made.

[**0065**] With the structure described above, less than desirable behavior can occur in a particular situation. The situation is when the RAMset is full and a clean operation is performed to write its data to the associated page of memory to make room for additional local variables in the RAMset. The cleaning process takes time and consumes power. A new method is invoked and then uses the newly mapped RAMset for its local variables. Returning from this method to the prior method entails flushing the RAMset and bringing the previously saved local variable data back into the RAMset from memory. This flushing and retrieval from memory